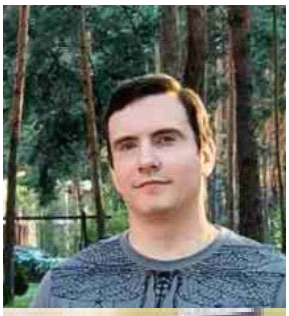
УДК 004.422 **DOI: 10.33744/0365-8171-2023-113.2-171-179**
UDC 004.422

ВИКОРИСТАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ ОЦІНКИ СКЛАДНОСТІ ЗАПИТУ У ФОРМАТІ JSON

USING NEURAL NETWORKS TO EVALUATE THE COMPLEXITY OF A JSON-FORMATTED QUERY



Сисоєв Ілля Костянтинович, *Національний транспортний Університет, аспірант кафедри інформаційних систем і технологій e-mail: i.sisoev.w@gmail.com*

orcid id: https://orcid.org/0000-0003-4823-9179

Гавриленко Валерій Володимирович, *Національний транспортний університет, завідувач кафедри інформаційних систем і технологій, доктор фізико-математичних наук, професор*
e-mail: vvgavrilenko1953@gmail.com

orcid id: https://orcid.org/0000-0001-9682-4204

Ковальчук Оксана Петрівна, *Національний транспортний університет, старший викладач кафедри інформаційних систем і технологій*
e-mail: kovalchukoksana30@gmail.com

orcid id: https://orcid.org/0000-0001-9456-8438

Анотація. В роботах [1, 2, 3, 4] було представлене описання багаторівневої системи балансування навантаження, на одному з рівнів якої пропонується використання технологій машинного навчання для аналізу вхідних запитів і прогнозування їх ресурсоємності. У даній статті розглядатиметься використання нейронних мереж для оцінки складності запиту у форматі JSON як з теоретичної, так і з практичної сторони. Проводиться теоретичний опис нейронних мереж, їх складових та особливостей, та розкривається питання складності запиту у форматі JSON.

Також пропонується системний підхід до оцінки та порівняння обчислювальної складності рівнів нейронної мережі в тестовій обробці сигналів JSON. Пов'язуються показники складності програмного

забезпечення та апаратного забезпечення, визначаючи їх як гіперпараметри шарів нейромережі. У роботі пояснюється, як обчислити метрики для прямого та повторюваного рівнів, а також визначається, у якому випадку має використовуватися певна метрика в залежності від програмного модуля, зорієнтованого на програмне або апаратне забезпечення. Ця робота може бути корисною для отримання різних рівнів (цілей) оцінки складності, пов'язаної із застосуванням нейронних мереж у тестовій обробці сигналів у реальному часі, а також для уніфікації оцінки обчислювальної складності.

Ключові слова: нейронні мережі, JSON, балансування, розробка алгоритму.

Вступ. У багаторівневій системі балансування навантаження поєднання нейронної мережі та JSON є важливим, оскільки це поєднання текстового опису системи та нейронної мережі створює адаптовані під сучасні вимоги програми, додатки тощо. Розглянемо їх поокремо, щоб зрозуміти всі особливості, функції та деталі, які пов'язують їх разом, так як разом вони створюють технологічний ефект.

Постанова задачі. Нейронні мережі – це набір алгоритмів, створених за зразком людського мозку, призначених для розпізнавання шаблонів. Вони інтерпретують сенсорні дані за допомогою свого роду машинного сприйняття, маркування або кластеризації вхідних даних. Шаблони, які вони розпізнають, є числовими, містяться у векторах, у які мають бути переведені всі дані реального світу, будь то зображення, звук, текст або часові ряди.

Нейронні мережі допомагають нам кластеризувати та класифікувати дані. Можемо розглядати нейронні мережі як підпрограми кластеризації та класифікації даних, які ви зберігаєте та керуєте ними. Вони допомагають групувати немарковані дані відповідно до подібності серед прикладів вхідних даних і класифікують дані, коли мають позначений набір даних для навчання. Нейронні мережі також можуть виділяти функції, які передаються іншим алгоритмам для кластеризації та класифікації, тому ви можете розглядати глибокі нейронні мережі як компоненти більш складних програм машинного навчання, що включають алгоритми навчання з підкріпленням, класифікацією та регресією.

JSON – це популярний формат текстових даних, який використовується для обміну даними у сучасних веб- та мобільних додатках. Крім того, JSON використовується для зберігання неструктурованих даних у файлах журналів або баз даних NoSQL, таких як Microsoft Azure Cosmos DB. Багато веб-служб REST повертають результати у форматі JSON або приймають дані у форматі JSON. Наприклад, більшість служб Azure, таких як пошук Azure, служба сховища Azure та Azure Cosmos DB, мають кінцеві точки REST, які повертають або приймають формат JSON. JSON – це також основний формат обміну даними між веб-сторінками та веб-серверами за допомогою дзвінків AJAX.

Функції JSON, які з'явилися у SQL Server 2016, дозволяють об'єднати принципи NoSQL та реляційних баз даних в одній базі даних. Тепер ви можете поєднувати в одній таблиці класичні реляційні стовпці зі стовпцями, які містять документи у форматі тексту JSON, аналізувати та імпортувати документи JSON у реляційні структури або форматувати реляційні дані у текст JSON.

Поєднання нейронних мереж та JSON є одним з корисних процесів, оскільки це основа під все, що ми можемо спостерігати на сьогоднішній час, додатки, веб-сторінки, боти та інше.

Мета статті. Мета даної статті – це теоретично описати всі складові, функції та особливості використання нейронних мереж для оцінки складності запиту у форматі JSON.

Виклад основного матеріалу. В інформаційних технологіях штучна нейронна мережа – це система апаратного та/або програмного забезпечення, що створена за принципом роботи нейронів у мозку людини. Штучні нейронні мережі, які також називають просто нейронними мережами, є різновидом технологій глибокого навчання, які також підпадають під егіду штучного інтелекту.

Комерційне застосування цих технологій, як правило, зосереджено на вирішенні проблем обробки складних сигналів або розпізнаванні зображень [5]. Приклади важливих комерційних застосувань з 2000 року включають розпізнавання рукописного тексту для обробки чеків, транскрипцію мовлення в текст, аналіз даних розвідки нафти, прогнозування погоди, розпізнавання облич тощо.

Історія штучних нейронних мереж сягає корінням у перші дні комп'ютерної техніки. У 1943 році математики Уоррен МакКаллох і Уолтер Піттс створили електричну систему, призначену для наближення функціонування людського мозку, яка керувала простими алгоритмами [6].

Лише приблизно в 2010 році дослідження знову пожвавилося. Тенденція big data, коли компанії накопичують величезні масиви даних, і паралельні обчислення дали додаткам дані для навчання та обчислювальні ресурси, необхідні для роботи складних штучних нейронних мереж. У 2012 році нейронна мережа змогла перевершити продуктивність людини в завданні розпізнавання зображень у рамках конкурсу ImageNet. Відтоді інтерес до штучних нейронних мереж різко зріс, а технологія продовжує вдосконалюватися.

Штучна нейронна мережа зазвичай включає велику кількість процесорів, що працюють паралельно та розташовані на різних рівнях. Перший рівень отримує необроблену вхідну інформацію – аналогічно зоровим нервам у обробці зору людини. Кожен наступний рівень отримує не вхідні дані, а вихідні дані від попереднього рівня – так само нейрони, розташовані далі від зорового нерва, отримують сигнали від тих, хто ближче до нього. Останній рівень створює вихідні дані системи.

Для кожного вузла обробки існує специфічна йому сфера знань, включаючи ту, що була закладена в нього спочатку, та будь-які правила, які він отримав в процесі обробки даних. Рівні взаємопов'язані, що означає, що кожен вузол рівня n буде під'єднаний до багатьох вузлів рівня (n-1) його входів і рівня (n+1) і надає вхідні дані для цих вузлів. У вихідному шарі може бути один або декілька вузлів, з яких можна прочитати отриману ним відповідь [7].

Штучні нейронні мережі відрізняються адаптивністю, що означає, що вони змінюються в міру того, як навчаються під час початкового навчання, а наступні запуски надають більше інформації про світ. Основна модель навчання зосереджена на зважуванні вхідних потоків, тобто кожен вузол зважує важливість вхідних даних від кожного зі своїх попередників. Вхідні дані, які сприяють отриманню правильних відповідей, мають вищу вагу.

Коли ми говоримо про нейронні мережі та JSON, варто сформувати їх у такому поясненні.

Позначення об'єктів JavaScript (JSON – JavaScript Object Notation) – стандартний текстовий формат для представлення структурованих даних на основі синтаксису об'єкта JavaScript. Зазвичай він використовується для передачі даних у веб-додатках (наприклад, для виправлення деяких даних із клієнтом сервера, таким чином, щоб вони могли відображатися на веб-сторінці).

JSON представляє собою рядок, формат якого дуже схожий на буквений формат об'єкта JavaScript. Ви можете включити одні й ті ж базові типи даних усередині JSON, так само як і в стандартному об'єкті JavaScript – рядки, числа, масиви, булеви та інші об'єктні літерали. Це дозволяє побудувати ієрархію даних, наприклад:

```
{
 "squadName": "Super hero squad",
 "homeTown": "Metro City",
 "formed": 2016,
 "secretBase": "Super tower",
 "active": true,
 "members": [
  {
   "name": "Molecule Man",
   "age": 29,
   "secretIdentity": "Dan Jukes",
   "powers": [
    "Radiation resistance",
    "Turning tiny",
    "Radiation blast"
   ]
```

```
    }
  ]
}
```

Нейронові мережі у використанні оцінки запитів JSON – це простий формат файлу для ієрархічного опису даних.

Keras (до прикладу) надає можливість описати будь-яку модель у форматі JSON за допомогою функції to_json(). Це можна зберегти у файл і пізніше завантажити за допомогою функції model_from_json(), яка створить нову модель із специфікації JSON [8].

Ваги зберігаються безпосередньо з моделі за допомогою функції save_weights() і пізніше завантажуються за допомогою симетричної функції load_weights().

У наведеному нижче прикладі тренується та оцінюється проста модель на наборі даних зареєстрованого автотранспорту України. Потім модель перетворюється у формат JSON і записується в model.json у локальному каталозі. Ваги мережі записуються в model.h5 у локальному каталозі.

Дані моделі та ваги завантажуються зі збережених файлів і створюється нова модель. Важливо скомпілювати завантажену модель перед її використанням. Це робиться для того, щоб прогнози, зроблені за допомогою моделі, могли використовувати відповідні ефективні обчислення з серверної частини Keras [9].

Модель оцінюється таким же чином, виконуючи наступну програму:

```
# MLP for Ukraine transport Dataset Serialize to JSON and HDF5
from tensorflow.keras.models import Sequential, model_from_json
from tensorflow.keras.layers import Dense
import numpy
import os
# fix random seed for reproducibility
numpy.random.seed(7)
# load Ukraine transport dataset
dataset = numpy.loadtxt("Ukraine-transport-diabetes.csv", delimiter=",")
# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
# Fit the model
model.fit(X, Y, epochs=150, batch_size=10, verbose=0)
# evaluate the model
scores = model.evaluate(X, Y, verbose=0)
print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))

# serialize model to JSON
model_json = model.to_json()
with open("model.json", "w") as json_file:
    json_file.write(model_json)
# serialize weights to HDF5
```

```
        model.save_weights("model.h5")
        print("Saved model to disk")

        # later...

        # load json and create model
        json_file = open('model.json', 'r')
        loaded_model_json = json_file.read()
        json_file.close()
        loaded_model = model_from_json(loaded_model_json)
        # load weights into new model
        loaded_model.load_weights("model.h5")
        print("Loaded model from disk")

        # evaluate loaded model on test data
        loaded_model.compile(loss='binary_crossentropy', optimizer='rmsprop',
metrics=['accuracy'])
        score = loaded_model.evaluate(X, Y, verbose=0)
        print("%s: %.2f%%" % (loaded_model.metrics_names[1], score[1]*100))
```

Ваші результати можуть відрізнятися з огляду на стохастичну природу алгоритму нейрону запиту формату JSON, на процедури оцінювання або на різницю в чисельній точності. Варто зазначити, що використання нейронних мереж для оцінки складності запиту у форматі JSON – це тема, яка досліджується, оскільки їх взаємодія та використання є мало дослідженою в області інформаційних технологій [10].

Підтримка JSON у SQL Server та базі даних SQL Azure дозволяє об'єднати принципи NoSQL та реляційних баз даних, які мають у собі нейронну мережу, легко перетворювати реляційні дані на частково структуровані і навпаки. Однак, JSON не замінює існуючі реляційні моделі. Нижче наведено деякі конкретні варіанти використання з перевагами підтримки JSON у SQL Server та базі даних SQL:

• Розгляд можливості денормалізації моделі даних із полями JSON замість кількох дочірніх таблиць.

• Збереження інформації про продукти, використовуючи в денормалізованій моделі безліч атрибутів змінних для забезпечення гнучкості.

• Завантаження та аналіз даних журналу, які зберігаються у вигляді JSON-файлів, використовуючи всі можливості мови Transact-SQL.

• Легке перетворення реляційних даних з бази даних у формат JSON, використовуваний інтерфейсами REST API, які підтримують ваш веб-сайт [11].

Отже, перш за все, ці нейронні мережі здатні виявляти приховані структури в немаркованих, неструктурованих даних, які становлять переважну більшість даних у світі. Іншим словом, для неструктурованих даних є необроблені носії, тобто зображення, тексти, відео та аудіозаписи. Таким чином, одна з проблем, яку найкраще вирішує глибоке навчання, полягає в обробці та кластеризації світових необроблених, немаркованих носіїв, розпізнанні подібності та аномалій у даних, які жодна людина не організувала в реляційній базі даних і не називала їх.

Наприклад, глибоке навчання може взяти мільйон зображень і згрупувати їх відповідно до їх схожості: легкові автомобілі в одну групу, вантажні в іншу, а в третій всі фотографії інших видів автотранспорту. Це основа так званих розумних фотоальбомів.

Тепер застосуйте ту саму ідею до інших типів даних: глибоке навчання може кластирувати необроблений текст, наприклад відгуки на стан автомобільних доріг. Відгуки, наповнені скаргами, можуть групуватися в одному секторі векторного простору, тоді як позитивні або повідомлення спаму

можуть групуватися в інших секторах розподілу. Це основа різних фільтрів повідомлень і може використовуватися в управлінні взаємовідносинами з клієнтами (CRM). Те саме стосується голосових повідомлень [12].

У часових рядах дані можуть групуватися навколо нормальної поведінки та аномальної поведінки агрегатів автомобіля. Якщо дані часових рядів генеруються за допомогою бортового комп'ютера авто, це дасть уявлення про його стан; якщо часовий ряд генерується з даних, що надходять з конкретних агрегатів автомобіля, його можна використовувати для запобігання суттєвим поломкам та попередити пошкодження агрегатів.

Мережі глибокого навчання виконують автоматичне виділення функцій без втручання людини, на відміну від більшості традиційних алгоритмів машинного навчання. Враховуючи те, що вилучення функцій – це завдання, на виконання якого команди спеціалістів із обробки даних можуть витратити роки, глибоке навчання – це спосіб обійти клопіт обмежених експертів [13]. Це збільшує повноваження невеликих наукових груп даних, які за своєю природою не масштабуються.

Під час навчання на немаркованих даних кожен рівень вузла в глибокій мережі вивчає функції автоматично, постійно намагаючись реконструювати вхідні дані, з яких він бере свої зразки, намагаючись мінімізувати різницю між припущеннями мережі та розподілом ймовірностей самих вхідних даних. Обмежені машини Больцмана, наприклад, створюють так звані реконструкції таким чином, що можуть самонавчатися без вчителя за допомогою алгоритму зворотного розподілення помилки.

У процесі нейронні мережі вчаться розпізнавати кореляції між певними релевантними функціями та оптимальними результатами – вони встановлюють зв'язки між сигналами ознак і тим, що вони представляють, будь то повна реконструкція чи дані з мітками [14].

Мережа глибокого навчання, навчена на мічених даних, може бути застосована до неструктурованих даних, надаючи їй доступ до набагато більшої кількості вхідних даних, ніж мережі машинного навчання. Це рецепт вищої продуктивності: чим з більшою кількістю даних може працювати мережа, тим точнішою вона буде. Погані алгоритми, навчені на великій кількості даних, можуть перевершити хороші алгоритми, навчені на дуже невеликій кількості даних. Здатність глибокого навчання обробляти величезну кількість непозначених даних і вчитися на них дає алгоритму явну перевагу перед попередніми алгоритмами.

Мережі глибокого навчання закінчуються вихідним рівнем: логістичним або JSON класифікатором, який призначає ймовірність певного результату або мітки. Ми називаємо це прогнозуванням, але це прогнозування в широкому сенсі. Отримавши необроблені дані у формі зображення, мережа глибокого навчання може вирішити, наприклад, що вхідні дані з імовірністю 90 відсотків представляють автомобіль певного типу.

Висновок. В статті показано, що застосовуючи нейронні мережі можна достатньо просто оцінити складність запиту. Тобто при масштабуванні і достатньому об'єму даних для навчання цю технологію можна застосувати в адаптивному алгоритмі балансування навантаження в додатках з розгалуженим обчисленням.

Перелік посилань

1. Сисоєв І.К. Адаптивний алгоритм балансування навантаження в додатках з використанням технології контейнеризації / І.К. Сисоєв, В.В. Гавриленко // Системи управління, навігації та зв′язку. Збірник наукових праць. – Полтава, 2022, вип. 1 (67). – С.81-83.

2. Гавриленко В.В. Проектування автомасштабованих високонавантажених додатків / В.В. Гавриленко, І.К. Сисоєв // VII Міжнародна науково-технічна конференція "Проблеми інформатизації", Харків, 2019. – С.15.

3. Сисоєв І.К. Адаптивний алгоритм балансування навантаження в додатках із використанням технології контейнеризації. / І.К. Сисоєв, В.В. Гавриленко // Матеріали VIII Міжнародної науково-технічної Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення

систем керування організаційно-технічними та технологічними комплексами», 26 листопада 2021. – К.: НУХТ, 2021. – С.272.

4. Гавриленко В.В. Управління контейнерами високонавантажених додатків в іт-системах / В.В. Гавриленко, І.К. Сисоєв // ІІІ Всеукраїнська науково-технічна конференція "Проблеми інфокомунікацій". Полтава-Київ-Харків-Мінськ, 19 листопада 2019 р. http://conf.itm.nupp.edu.ua/index.php/pi/3pi

5. Гавриленко В.В., Іванченко Г.Ф., Шевченко Г.Є. Теорія розпізнавання образів. Навчальний посібник для студ. НТУ, які навч. за напр. «Комп'ютерні науки». – К.: НТУ, 2015. – 76 с.

6. Matt Crane and Jimmy Lin. 2017. An exploration of serverless architectures for information retrieval. In Proceedings of the 3rd ACM International Conference on the Theory of Information Retrieval (ICTIR 2017), pages 241-244, Amsterdam, The Netherlands.

7. Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swami Sivasubramanian, Peter Vosshall, and Werner Vogels. 2007. Dynamo: Amazon's highly available key-value store. In Proceedings of the 21st ACM Symposium on Operating Systems Principles (SOSP 2007), pages 205-220, Stevenson, Washington.

8. Yoon Kim. 2014. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pages 1746-1751, Doha, Qatar.

9. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv:1301.3781.

10. Christopher Olston, Noah Fiedel, Kiril Gorovoy, Jeremiah Harmsen, Li Lao, Fangwei Li, Vinu Rajashekhar, Sukriti Ramesh, and Jordan Soyke. 2017. TensorFlow-Serving: Flexible, high-performance ML serving. In Workshop on ML Systems at NIPS 2017.

11. Jinfeng Rao, Hua He, and Jimmy Lin. 2017. Experiments with convolutional neural network models for answer selection. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1217-1220. ACM.

12. Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In Proceedings of the 38th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015), pages 373-382, Santiago, Chile.

13. Сисоєв І.К. Перспективи алгоритмічної мови python в опануванні студентами дисциплін машинного навчання / І.К. Сисоєв, В.В. Гавриленко, О.А. Шумейко, Н.В. Рудоман, В.В. Донець // Вісник Національного транспортного університету. Серія «Технічні науки». Науково-технічний збірник. – К.: НТУ, 2022. – Вип. 3 (53). – С.337-343. DOI: 10.33744/2308-6645-2022-3-53-337-343.

14. Гавриленко В.В. Використання штучних нейронних мереж для оцінки складності запитів / В.В. Гавриленко, І.К. Сисоєв, А.В. Ляшко // Матеріали V Міжнародної науково-практичної конференції "Сучасні тенденції розвитку інформаційних систем і телекомунікаційних технологій". – К.: НУХТ, 2023. – С.37.

USING NEURAL NETWORKS TO EVALUATE THE COMPLEXITY OF A JSON-FORMATTED QUERY

Sisoev Illya K., National Transport University, PhD-student of the Department of Information Systems and Technologies, Kyiv 01010, Omelyanovich-Pavlenko Str. 1, phone: +380958650637, e-mail: i.sisoev.w@gmail.com, orcid id: https://orcid.org/0000-0003-4823-9179

Gavrilenko Valeriy V., National Transport University, Doctor of Physical and Mathematical Sciences, Professor, Head of Department of Information Systems and Technologies, Kyiv 01010, Omelyanovich-Pavlenko Str.1, phone: +380503806406, e-mail: vvgavrilenko1953@gmail.com, orcid id: https://orcid.org/0000-0001-9682-4204

Kovalchuk Oksana P., National Transport University, Senior Lecturer of the Department of Information Systems and Technologies, Kyiv 01010, Omelyanovich-Pavlenko Str. 1, phone: +380968349605, e-mail: kovalchukoksana30@gmail.com, orcid id: https://orcid.org/ 0000-0001-9456-8438

Abstract. In the previous articles [1, 2, 3, 4], a description of a multilevel load balancing system was presented, where one of the levels proposes the use of machine learning technologies for analyzing input queries and predicting their resource requirements. This article focuses on the use of neural networks for assessing the complexity of JSON-formatted queries from both theoretical and practical perspectives. The theoretical description of neural networks, their components, and peculiarities is provided, and the issue of query complexity in JSON format is explored.

Additionally, a systematic approach is proposed for evaluating and comparing the computational complexity of neural network levels in the test processing of JSON signals. The connection between software and hardware complexity indicators is established by defining them as hyperparameters of the neural network layers. The paper explains how to compute metrics for the forward and recurrent levels and determines the specific metrics to be used based on whether the focus is on software or hardware-oriented modules. This work can be valuable for obtaining different levels (goals) of complexity assessment related to the application of neural networks in real-time signal processing and for standardizing the evaluation of computational complexity.

Overall, this research provides insights into the utilization of neural networks for assessing query complexity in the JSON format, offering a systematic approach to evaluating computational complexity in the context of neural network levels.

Keywords: neural networks, JSON, balancing, algorithm development.

References

1. Sysoev I.K. Adaptive load balancing algorithm in applications using containerization technology / I.K. Sysoev, V.V. Havrylenko // Management, navigation and communication systems. Collection of scientific works. – Poltava, 2022, issue 1 (67). – P.81-83.

2. Gavrylenko V.V. Design of self-scaling highly loaded applications / V.V. Gavrylenko, I.K. Sysoev // VII International Scientific and Technical Conference "Problems of Informatization", Kharkiv, 2019. – P.15.

3. Sysoev I.K. Adaptive load balancing algorithm in applications using containerization technology. / I.K. Sysoev, V.V. Gavrylenko // Materials of the VIII International scientific and technical Internet conference "Modern methods, information, software and technical support of management systems of organizational, technical and technological complexes", November 26, 2021. – K.: NUHT, 2021. – P.272.

4. Gavrylenko V.V. Container management of highly loaded applications in IT systems / V.V. Gavrylenko, I.K. Sysoev // III All-Ukrainian Scientific and Technical Conference "Problems of Information Communications". Poltava-Kyiv-Kharkiv- Minsk, November 19, 2019. http://conf.itm.nupp.edu.ua/index.php/pi/3pi

5. Gavrylenko V.V., Ivanchenko G.F., Shevchenko G.E. Pattern recognition theory. Study guide for students. NTU, which teaches for example "Computer Science". – K.: NTU, 2015. – 76 p.

6. Matt Crane and Jimmy Lin. 2017. An exploration of serverless architectures for information retrieval. In Proceedings of the 3rd ACM International Conference on the Theory of Information Retrieval (ICTIR 2017), pages 241-244, Amsterdam, The Netherlands.

7. Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swami Sivasubramanian, Peter Vosshall, and Werner Vogels. 2007. Dynamo: Amazon's highly available key-value store. In Proceedings of the 21st ACM Symposium on Operating Systems Principles (SOSP 2007), pages 205-220, Stevenson, Washington.

8. Yoon Kim. 2014. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pages 1746-1751, Doha, Qatar.

9. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv:1301.3781.

10. Christopher Olston, Noah Fiedel, Kiril Gorovoy, Jeremiah Harmsen, Li Lao, Fangwei Li, Vinu Rajashekhar, Sukriti Ramesh, and Jordan Soyke. 2017. TensorFlow-Serving: Flexible, high-performance ML serving. In Workshop on ML Systems at NIPS 2017.

11. Jinfeng Rao, Hua He, and Jimmy Lin. 2017. Experiments with convolutional neural network models for answer selection. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1217-1220. ACM.

12. Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In Proceedings of the 38th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015), pages 373-382, Santiago, Chile.

13. Sysoev I.K. Perspectives of the algorithmic python language in mastering by students of machine learning disciplines / I.K. Sysoev, V.V. Havrylenko, O.A. Shumeiko, N.V. Rudoman, V.V. Donets // Bulletin of the National Transport University. Series "Technical Sciences". Scientific and technical collection. – K.: NTU, 2022. – Issue 3 (53). – P.337-343. DOI: 10.33744/2308-6645-2022-3-53-337-343.

14. Gavrylenko V.V. The use of artificial neural networks to assess the complexity of requests / V.V. Gavrylenko, I.K. Sysoev, A.V. Lyashko // Proceedings of the V International scientific and practical conference "Modern trends in the development of information systems and telecommunication technologies". – K.: NUHT, 2023. – P.37.

Надійшла до редакції 04.06.2023.